



# Software Diversification for WebAssembly

JAVIER CABRERA-ARTEAGA

Doctoral Thesis in Computer Science  
Supervised by  
Benoit Baudry and Martin Monperrus

Stockholm, Sweden, March 2024

KTH Royal Institute of Technology  
School of Electrical Engineering and Computer Science  
Division of Software and Computer Systems  
TRITA-EECS-AVL-2020:4 SE-10044 Stockholm  
ISBN 100- Sweden

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges  
till offentlig granskning för avläggande av Teknologie doktorexamen i elektroteknik  
i .

© Javier Cabrera-Arteaga , March 7th 2024

Tryck: Universitetsservice US AB

## Abstract

WebAssembly has become the fourth officially recognized web language, allowing web browsers to adapt native applications for Web. Moreover, WebAssembly has developed into a critical component of backend scenarios such as edge computing and cloud computing. Nowadays, WebAssembly is used in a wide range of applications, including web browsers, blockchain, and cloud computing. While security was a primary focus in its design, WebAssembly remains vulnerable to attacks, including side-channels and memory corruption. In addition, WebAssembly has been exploited to transport malware, especially in instances of browser cryptojacking. Remarkably, the predictability of the WebAssembly ecosystem, including its users and the programs it hosts, is exceedingly high. This predictability can exacerbate the impact of a vulnerability within these ecosystems. For example, a flaw in a web browser, instigated by a faulty WebAssembly program, could potentially affect millions of users.

This thesis aims to enhance the security of the WebAssembly ecosystem through the introduction of methods and tools for Software Diversification. Software Diversification is a strategy designed to augment the cost of exploitation by rendering the software less predictable. By automatically generating numerous variants of a program, we can decrease predictability within ecosystems. These variants harden observable properties typically utilized to carry out attacks. For instance, we can generate variants of a program with diverse memory layouts and control-flow graphs, thereby strengthening code analysis, execution traces, and execution times. Yet, in the context of WebAssembly, Software Diversification has not been explored.

We present three pioneering tools to the community: CROW, MEWE, and WASM-MUTATE. Each tool is specifically designed to address a unique aspect of Software Diversification. Moreover, these tools synergistically enhance each other. We furnish empirical evidence that Software Diversification is applicable to WebAssembly programs in two distinct manners: Offensive and Defensive Software Diversification. Our investigation into Offensive Software Diversification in WebAssembly reveals potential avenues for improving the detection of WebAssembly malware. In contrast, our experiments in Defensive Software Diversification demonstrate that WebAssembly programs can be strengthened against side-channel attacks, specifically against the Spectre attack.

**Keywords:** WebAssembly, Software Diversification, Side-Channels, Moving Target Defense

## **Sammanfattning**

# LIST OF PAPERS

1. ***WebAssembly Diversification for Malware Evasion***  
**Javier Cabrera-Arteaga**, Tim Toady, Martin Monperrus, Benoit Baudry  
*Computers & Security, Volume 131, 2023, 17 pages*  
<https://www.sciencedirect.com/science/article/pii/S0167404823002067>
2. ***Wasm-mutate: Fast and Effective Binary Diversification for WebAssembly***  
**Javier Cabrera-Arteaga**, Nicholas Fitzgerald, Martin Monperrus, Benoit Baudry  
*Submitted to Computers & Security, under revision, 17 pages*  
<https://arxiv.org/pdf/2309.07638.pdf>
3. ***Multi-Variant Execution at the Edge***  
**Javier Cabrera-Arteaga**, Pierre Laperdrix, Martin Monperrus, Benoit Baudry  
*Moving Target Defense (MTD 2022), 12 pages*  
<https://dl.acm.org/doi/abs/10.1145/3560828.3564007>
4. ***CROW: Code Diversification for WebAssembly***  
**Javier Cabrera-Arteaga**, Orestis Floros, Oscar Vera-Pérez, Benoit Baudry, Martin Monperrus  
*Measurements, Attacks, and Defenses for the Web (MADWeb 2021), 12 pages*  
<https://doi.org/10.14722/madweb.2021.23004>
5. ***Superoptimization of WebAssembly Bytecode***  
**Javier Cabrera-Arteaga**, Shrinish Donde, Jian Gu, Orestis Floros, Lucas Satabin, Benoit Baudry, Martin Monperrus  
*Conference Companion of the 4th International Conference on Art, Science, and Engineering of Programming (Programming 2021), MoreVMs, 4 pages*  
<https://doi.org/10.1145/3397537.3397567>
6. ***Scalable Comparison of JavaScript V8 Bytecode Traces***  
**Javier Cabrera-Arteaga**, Martin Monperrus, Benoit Baudry  
*11th ACM SIGPLAN International Workshop on Virtual Machines and Intermediate Languages (SPLASH 2019), 10 pages*  
<https://doi.org/10.1145/3358504.3361228>

# ACKNOWLEDGEMENT

**TODO** W **TODO** O  
**TODO** Jury  
**TODO** C  
**TODO** F

# Contents

<b>List of Papers</b>	<b>iii</b>
<b>Acknowledgement</b>	<b>iv</b>
<b>Contents</b>	<b>1</b>
<b>I Thesis</b>	<b>4</b>
<b>1 Introduction</b>	<b>5</b>
1.1 Predictability in WebAssembly ecosystems . . . . .	8
1.2 Problems statements . . . . .	9
1.3 Software Diversification . . . . .	9
1.4 Summary of research papers . . . . .	10
<b>2 Background and state of the art</b>	<b>13</b>
2.1 WebAssembly . . . . .	13
2.1.1 From source code to WebAssembly . . . . .	14
2.1.2 WebAssembly’s binary format . . . . .	17
2.1.3 WebAssembly’s runtime . . . . .	18
2.1.4 WebAssembly’s control-flow . . . . .	20
2.1.5 Security and Reliability for WebAssembly . . . . .	21
2.1.6 Open challenges . . . . .	22
2.2 Software diversification . . . . .	23
2.2.1 Automatic generation of software variants . . . . .	24
2.2.2 Equivalence Checking . . . . .	26
2.2.3 Variants deployment . . . . .	27
2.2.4 Measuring Software Diversification . . . . .	28
2.2.5 Offensive or Defensive assessment of diversification . . . . .	29
2.3 Open challenges for Software Diversification . . . . .	30

<b>3</b>	<b>Automatic Software Diversification for WebAssembly</b>	<b>32</b>
3.1	CROW: Code Randomization of WebAssembly . . . . .	33
3.1.1	Enumerative synthesis . . . . .	34
3.1.2	Constant inferring . . . . .	35
3.1.3	Exemplifying CROW . . . . .	36
3.2	MEWE: Multi-variant Execution for WebAssembly . . . . .	38
3.2.1	Multivariant call graph . . . . .	39
3.2.2	Exemplifying a Multivariant binary . . . . .	39
3.3	WASM-MUTATE: Fast and Effective Binary Diversification for WebAssembly . . . . .	42
3.3.1	WebAssembly Rewriting Rules . . . . .	43
3.3.2	E-Graphs traversals . . . . .	44
3.3.3	Exemplifying WASM-MUTATE . . . . .	45
3.4	Comparing CROW, MEWE, and WASM-MUTATE . . . . .	47
3.4.1	Security applications . . . . .	50
<b>4</b>	<b>Assessing Software Diversification for WebAssembly</b>	<b>52</b>
4.1	Offensive Diversification: Malware evasion . . . . .	52
4.1.1	Cryptojacking defense evasion . . . . .	53
4.1.2	Methodology . . . . .	54
4.1.3	Results . . . . .	56
4.2	Defensive Diversification: Speculative Side-channel protection . . . . .	60
4.2.1	Threat model: speculative side-channel attacks . . . . .	61
4.2.2	Methodology . . . . .	61
4.2.3	Results . . . . .	63
<b>5</b>	<b>Conclusions and Future Work</b>	<b>69</b>
5.1	Summary of technical contributions . . . . .	69
5.2	Summary of empirical findings. . . . .	70
5.3	Future Work . . . . .	71
5.3.1	Improving WebAssembly malware detection via canonicalization . . . . .	71
5.3.2	Code pattern feedback-guided Diversification . . . . .	72
	<b>References</b>	<b>74</b>
<b>II</b>	<b>Included papers</b>	<b>88</b>
	WebAssembly Diversification for Malware Evasion	<b>90</b>
	Wasm-mutate: Fast and Effective Binary Diversification for WebAssembly	<b>91</b>



<i>CONTENTS</i>	3
CROW: Code Diversification for WebAssembly	<b>92</b>
Multi-Variant Execution at the Edge	<b>93</b>
Superoptimization of WebAssembly Bytecode	<b>94</b>
Scalable Comparison of JavaScript V8 Bytecode Traces	<b>95</b>

**Part I**

**Thesis**